# Aligning reads to a reference genome and BAM refinement Practical Handbook


**Summary**

During this practical you will

- map raw reads to a reference genome
- refine bam files
- visualise bam files


**Data Files**

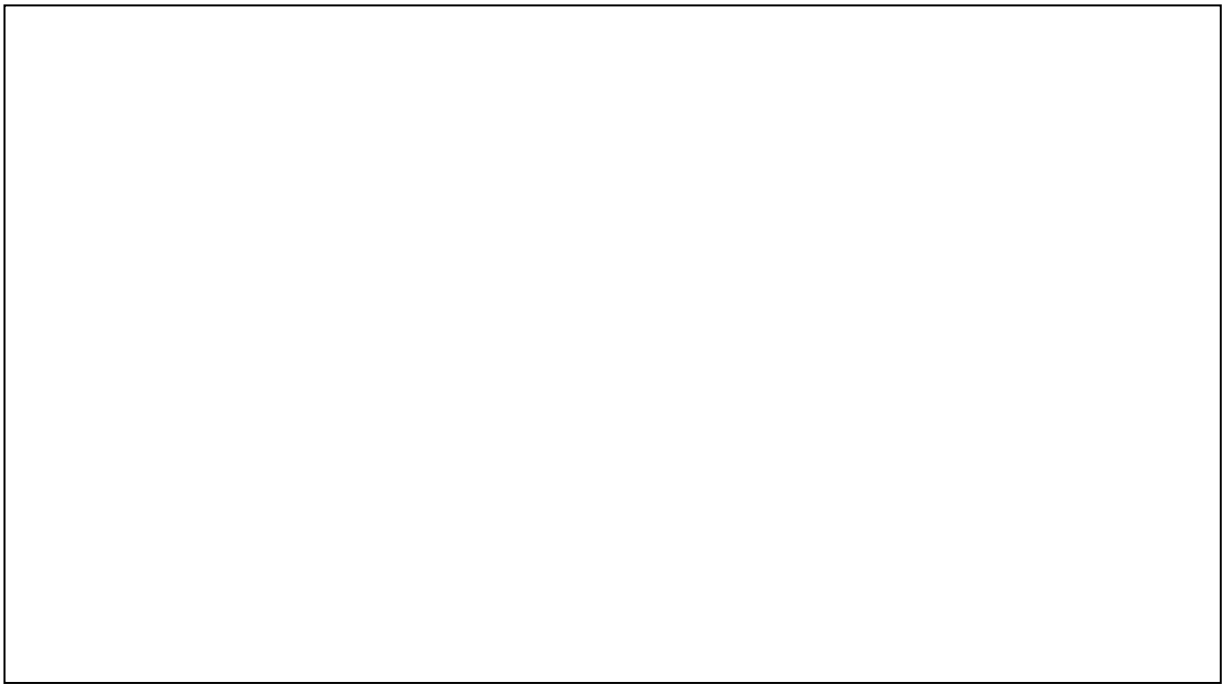
The data used here is a subset of re-sequencing data from *Saccharomyces cerevisciae* (from Thomas Keane, EBI).

Pipeline adapted from re-sequencing workflow by Joshua C. Randall, EBI.

And, finally, Kate Lee (BBASH, University of Leicester) wrote the original version of this handbook.

Characteristics of the experiment:

- Yeast genome: 12.5 Mbp; 16 chromosomes
- Whole genome sequencing
- Paired-end reads, 108bp, one library, 2 lanes



**Software Used:**

**Burrows-Wheeler Alignment tool (BWA)** maps sequencing reads to closely-related reference genomes. First an index is created of the reference and then a selection of algorithms can be used to align different types of read data. http://bio-bwa.sourceforge.net/bwa.shtml

**Picard** is a collection of java scripts to manipulate NGS data and formats. http://broadinstitute.github.io/picard/

**Samtools** is collections of utilities for manipulating sam /bam files. http://samtools.sourceforge.net/samtools.shtml

**Genome Analysis Toolkit (GATK)** software is designed for variant discovery and genotyping. http://www.broadinstitute.org/gatk/

**IGV** is a genome visualisation tool. https://www.broadinstitute.org/software/igv/download

**Vcftools** is a set of scripts to manipulate vcf files. http://vcftools.sourceforge.net/

**Getting the Data**

Move to your scratch area

**cd $CINECA_SCRATCH**

copy folder from teaching directory for use

**cp -r /pico/scratch/userexternal/cbatini0/day3 .**

move to new folder

**cd day3**

You should now be in a folder called day3 containing read data (lane1, lane2), a reference genome (Saccharomyces_cerevisiae.EF4.68.dna.toplevel.fa), co-ordinates of yeast mtDNA (mito.intervals), the pdf of the slides (day3_mapping_BAM_refinement_nov2015.pdf) and the pdf of this handbook (day3_mapping_BAM_refinement_handbook_nov2015.pdf).

Hint: you can use scp to copy files locally.

Check your location in the file directory using the **pwd** command (you should be in your scratch directory LOCATION: **/pico/scratch/userexternal/username/day3**)

Check the contents of the folder using the **ls** command.

**Create Index and dictionary files of the reference genome for samtools, bwa and picard**

Indices are necessary for quick access to specific information in very large files. Here we will create indices for the Saccharomyces reference genome for tools we will use downstream in the pipeline. For example the samtools index file, 'ref_name.fai', stores records of sequence identifier, length, the offset of the first sequence character in the file, the number of characters per line and the number of bytes per line.

LOCATION: **/pico/scratch/userexternal/username/day3**

As you generate each index look at the files created using the ls command

Samtools index

**module load autoload samtools**

**samtools faidx Saccharomyces_cerevisiae.EF4.68.dna.toplevel.fa**

bwa index:

**module load bwa**

**bwa index -a is Saccharomyces_cerevisiae.EF4.68.dna.toplevel.fa**

-a is Sets the algorithm to be used to construct a suffix array. This is suitable for databases smaller than 2GB.

Picard Dictionary:

module load autoload picard

java -jar /cineca/prod/applications/picard/1.119/binary/bin/CreateSequenceDictionary.jar R= Saccharomyces_cerevisiae.EF4.68.dna.toplevel.fa O= Saccharomyces_cerevisiae.EF4.68.dna.toplevel.dict

Name the extensions of the files (e.g. '.txt', '.sam') that have been created for

indices of

samtools:

bwa:

picard:

**Align reads to the Reference Genome using BWA**

BWA uses the burrows wheeler algorithm to compress data and efficiently parse the reference for sequence matches. Bwa mem is the latest bwa algorithm and is recommended for high-quality data as it is faster and more accurate.

LOCATION: **/pico/scratch/userexternal/username/day3**

Align reads using bwa mem

**bwa mem -M Saccharomyces_cerevisiae.EF4.68.dna.toplevel.fa lane1/s-7-1.fastq lane1/s-7-2.fastq > lane1.sam**

options used:

-M Mark shorter split hits as secondary (for Picard compatibility).

From https://www.biostars.org/p/97323/:

- with option -M it is flagged as a duplicate flag=256 ( not primary alignment ): will be ignored by most 'old' tools.
- without -M, a **split read** is flagged as 2048 ( supplementary alignment ) see http://picard.sourceforge.net/explain-flags.html. This flag is a recent addition to the SAM spec.

Other commonly used options include:

-t number of threads/processers to use – see PBS script at end of workbook

-p Assume the first input query file is interleaved paired-end FASTA/Q. See the command description for details.

-a Output all found alignments for single-end or unpaired paired-end reads. These alignments will be flagged as secondary alignments.

See bwa manual for more options.

Convert the new sam file to bam format (bam is a binary version of the sam format)

**samtools view -S -b lane1.sam -o lane1.bam**

options used:

-S Input is a sam file. If @SQ header lines are absent, the '-t' option is required.

-b Output a bam file.

-o output file

sort the bam (this adds the bam extension automatically!)

**samtools sort lane1.bam lane1_sorted**

Samtools sorts alignments by their leftmost chromosomal coordinates. (Can sort by read name instead using '–t' option.)

index the sorted bam for fast access

**samtools index lane1_sorted.bam**

**Can you guess the extension of this file? Check it in your folder… (use unix ls and options)**

Have a look at the header of your new bam file

**samtools view –H lane1_sorted.bam**

**How many chromosomes are present and which version of the SAM is it?**

**use unix command more on your SAM file and check what is after the header...**

**Align lane 2 data, convert to sam, sort and index using the samtools commands above, but changing the file names where appropriate.**

NOTE: you can use the **arrow keys** to move through commands you have issued in the terminal – file and folder names can then be easily changed in earlier commands. Try to save a copy of the new commands you use or use the **history** command to keep a record of what you have done.

CHECK you should now have aligned sorted and indexed files for both lanes.

lane1_sorted.bam
lane1_sorted.bam.bai
lane2_sorted.bam
lane2_sorted.bam.bai

Merge BAMs per library using picard MergeSamFiles

**java -jar /cineca/prod/applications/picard/1.119/binary/bin/MergeSamFiles.jar INPUT=lane1_sorted.bam INPUT=lane2_sorted.bam OUTPUT=library.bam**

Add read group header using picard AddOrReplaceReadGroups (please keep in mind that there is a way to do this during the alignment with bwa with the option -R)

**java -jar /cineca/prod/applications/picard/1.119/binary/bin/AddOrReplaceReadGroups.jar INPUT=library.bam OUTPUT=library_RG.bam RGID=1 RGLB=library RGPL=Illumina RGPU=lane1_2 RGSM=yeast**

| | |
|---|---|
| RGID (String) | Read Group ID Default value: 1. This option can be set to 'null' to clear the default value. |
| RGLB (String) | Read Group Library Required. |
| RGPL (String) | Read Group platform (e.g. illumina, solid) Required. |
| RGPU (String) | Read Group platform unit (eg. run barcode) Required. |
| RGSM (String) | Read Group sample name Required. |
| RGCN (String) | Read Group sequencing center name Default value: null. |
| RGDS (String) | Read Group description Default value: null. |
| RGDT (Iso8601Date) | Read Group run date Default value: null. |
| RGPI (Integer) | Read Group predicted insert size Default value: null. |
| RGPG (String) | Read Group program group Default value: null. |
| RGPM (String) | Read Group platform model Default value: null. |

Index and sort the merged bam file with read groups

**BAM refinement – local realignment and duplicate removal**

Local alignment with GATK

Indels in the data that are not present in the reference genome can cause small mis-alignments at the end of the reads. GATK's local re-alignment identifies the areas characterized by a high number of mis-matching bases and realigns the reads around it.

LOCATION: **/pico/scratch/userexternal/username/day3**

Load the GATK module

**module load autoload gatk/3.3.0**

We are using the Genome Anlaysis Toolkit (GenomeAnalysisTK.jar) to carry out local re-alignment.

The options for commands below are:

-I input bam file

-R reference genome

-T tool (RealignerTargetCreator and IndelRealigner are used below)

-o output file

1. RealignerTargetCreator: identifies regions that need re-alignment

**java -jar /cineca/prod/applications/gatk/3.3.0/jre--1.7.0_72/GenomeAnalysisTK.jar -I library_RG_sorted.bam -R Saccharomyces_cerevisiae.EF4.68.dna.toplevel.fa -T RealignerTargetCreator -o library_targets.intervals**

2. IndelRealigner: re-aligns target regions

**java -jar /cineca/prod/applications/gatk/3.3.0/jre--1.7.0_72/GenomeAnalysisTK.jar -I library_RG_sorted.bam -R Saccharomyces_cerevisiae.EF4.68.dna.toplevel.fa -T IndelRealigner -targetIntervals library_targets.intervals -o library_RG_sorted_lr.bam**

More options can be found in the documentation on the GATK website
http://www.broadinstitute.org/gatk/gatkdocs/org_broadinstitute_sting_gatk_walkers_indels_RealignerTargetCreator.html

Duplicate removal with picard

PCR duplicates may confound coverage estimates and amplify the effects of mis-calls.

LOCATION: **/pico/scratch/userexternal/username/day3**

Remove duplicates using picard MarkDuplicates

**java -jar /cineca/prod/applications/picard/1.119/binary/bin/MarkDuplicates.jar INPUT=library_RG_sorted_lr.bam OUTPUT=library_final.bam METRICS_FILE=dupl_metrics.txt**

Sort and index library_final.bam file

**BAM QC**

look at metrics file from bam refinement

**gedit dupl_metrics.txt &**

**What's the percentage of duplicated reads?**

Get samtools flagstat metrics

**samtools flagstat library.bam > library_raw_flagstat.txt**

**samtools flagstat library_final_sorted.bam > library_flagstat.txt**

**Note the differences between samtools flagstat output before and after refinement.**

Look at the coverage per position in mitochondria

**java -jar /cineca/prod/applications/gatk/3.3.0/jre--1.7.0_72/GenomeAnalysisTK.jar -T DepthOfCoverage -R Saccharomyces_cerevisiae.EF4.68.dna.toplevel.fa -I library_final_sorted.bam -o mito_coverage -L mito.intervals**

option used:

-L interval list for genes of interest

Look at average coverage

**gedit mito_coverage.sample_summary &**

**BAM visualisation**

We will use the java web start version of IGV.
Follow this link: https://www.broadinstitute.org/software/igv/download
Register, and you'll find the IGV Java Web start. Launch IGV with 750 MB.
Be patient when you use IGV.

**extract mtDNA from final BAM**
samtools view -o mito.bam library_final_sorted.bam Mito

**index the mito bam file**

**Do you know why we have used Mito to extract the mtDNA? Check your dictionary or your bam header.**

**Download the mito bam file, its index and the reference genome fast file locally.**

Hint: you can use scp to copy files locally.

You will see that the default reference genome loaded is Human hg19. Load your reference genome (check Genomes) and then your bam file (check File).

**What can you see in the IGV visualisation, that is not obvious in the**

**mito_coverage.sample_summary?**

**Plot coverage per position for the mtDNA with R**

We are now going to use R to create a plot showing the coverage for each position of the mitochondrial DNA. We will start from the file mito_coverage. Have a look at it in gedit.

```
more mito_coverage
```

**Start R**

```
module load r
R
```

Once in R, import the file mito_coverage as a table, create a new column containing the position on the Mito and plot this column and the coverage for our sample in a simple x,y plot.

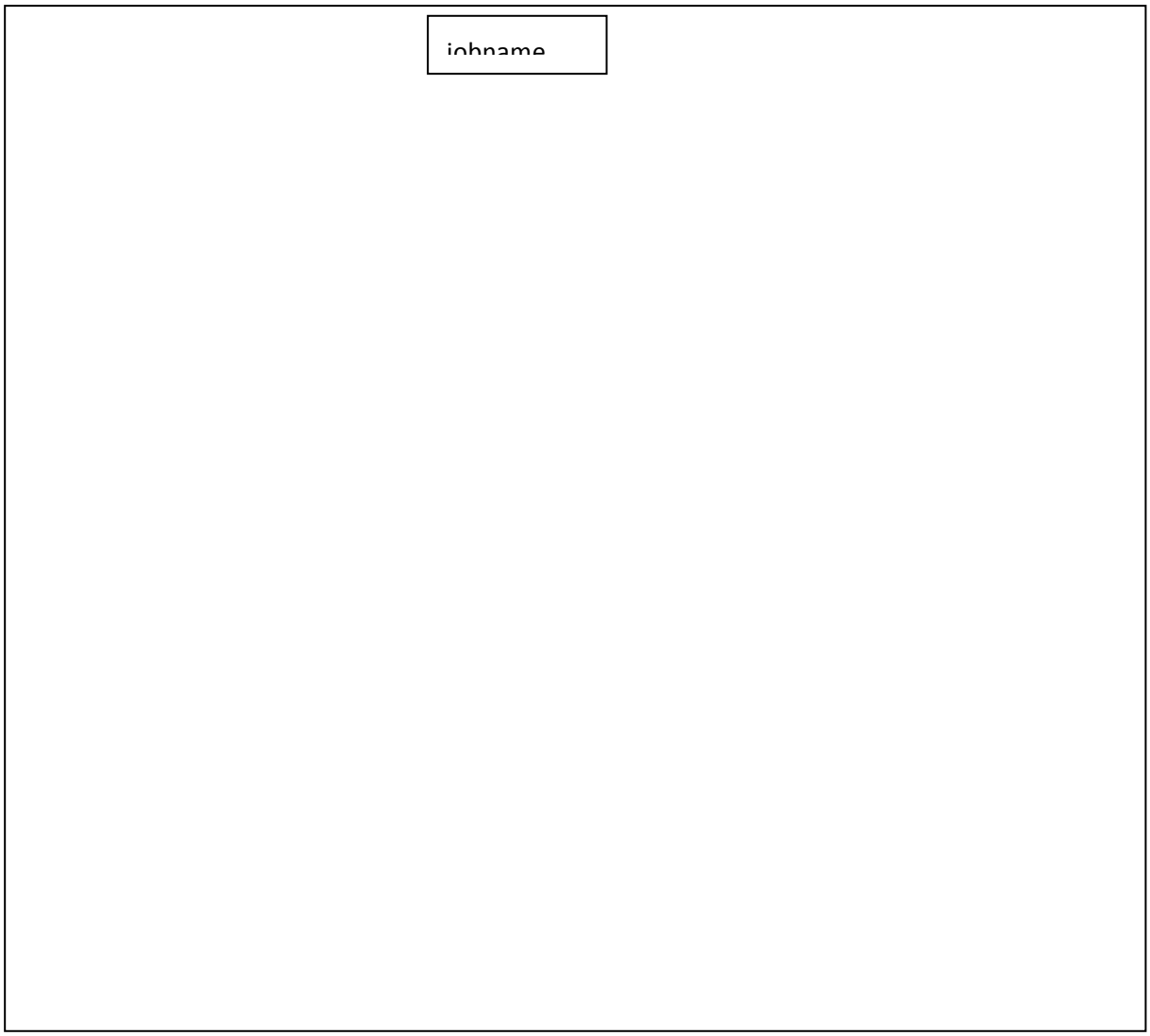
```
#import the table, specifying tab as the separator among columns and defining the first row as a header
data <- read.table("/pico/scratch/userexternal/cbatini0/day3/mito_coverage", sep="\t", header=T)
#check the names of the columns
names(data)
#define your first column as old_col
```

**old_col<-data$Locus OR old_col<-data[,1]**
#create a new column containing the old_col values minus “Mito:” (this will leave only the position, which can be used for the plot)
**new_col<-gsub("Mito:","",as.character(old_col))**
#create a new column in the dataframe “data” and call it “pos”
**data["pos"]<-new_col**
#plot the values in “pos” on the x and the values in “Depth_for_yeast” on the y, specifying that you want a line
**plot(data$pos,data$Depth_for_yeast,type="l") OR**
**plot(data[,5],data[,4],type="l")**

## EXTRA

### Automating your pipeline

Once you have established a pipeline for your data that you are happy with, you can run it as a job on your local server. This is especially useful for large datasets, when some commands can be threaded onto multiple processors, reducing the run-time. Please note that you will still need to check your data at each stage to ensure the process is running smoothly.

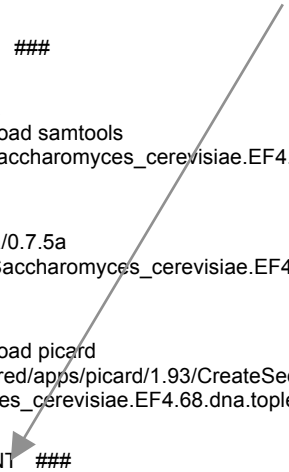



PBS script for Indexing Reference and Aligning data

```
#PBS -N bwa
#PBS -l walltime=02:00:00
#PBS -l vmem=10gb
#PBS -m bea
#PBS -M email address
#PBS –A train_Elixi15
#PBS -l nodes=1:ppn=8

cd $PBS_O_WORKDIR
module load profile/advanced
```

Requesting a number of processors, allows threading in commands below



```
###   INDEXING   ###

# samtools index
module load autload samtools
samtools faidx Saccharomyces_cerevisiae.EF4.68.dna.toplevel.fa

# bwa index
module load bwa/0.7.5a
bwa index -a is Saccharomyces_cerevisiae.EF4.68.dna.toplevel.fa

# picard index
module load autload picard
java -jar /cm/shared/apps/picard/1.93/CreateSequenceDictionary.jar R=Saccharomyces_cerevisiae.EF4.68.dna.toplevel.fa
O=Saccharomyces_cerevisiae.EF4.68.dna.toplevel.dict

###   ALIGNMENT   ###

# bwa mem alignment
bwa mem  -M –t 8 Saccharomyces_cerevisiae.EF4.68.dna.toplevel.fa lane1/s-7-1.fastq lane1/s-7-2.fastq > lane1.sam

# convert sam file to bam file
samtools view -S -b lane1.sam -o lane1.bam

# sort and index file
samtools sort lane1.bam lane1_sorted
samtools index lane1_sorted.bam

# lane 2 data
bwa mem  -M –t 8 Saccharomyces_cerevisiae.EF4.68.dna.toplevel.fa lane2/s-7-1.fastq lane2/s-7-2.fastq > lane2.sam
samtools view -S -b lane2.sam -o lane2.bam
samtools sort lane2.bam lane2_sorted
samtools index lane2_sorted.bam
```

submit your script using **qsub scriptname.pbs**